METHODS AND SYSTEMS FOR USER MEDIA INTEROPERABILITY WITH DATA INTEGRITY

CROSS-REFERENCE TO RELATED PATENT APPLICATIONS

This application is a continuation-in-part application of copending U.S. Patent Application No. 10/732,412, filed December 11, 2003, which claims the benefit of U.S. Provisional Patent Application No. 60/432,235, filed December 11, 2002, and U.S. Provisional Patent Application No. 60/439,444, filed January 13, 2003, which applications are hereby incorporated by reference herein in their entirety.

FIELD OF THE INVENTION

The present invention is related to interoperability of user media with different service providers, and more particularly to smart card interoperability and data integrity.

BACKGROUND ART

There is presently an increasing trend towards the use of smart cards for personal services. Smart cards are convenient and portable for automatically obtaining services from a variety of different vendors. The increased demand may result in the accumulation of a large number of smart cards that are issued by different service providers for their corresponding services. The accumulation of smart cards for each user may present a significant inconvenience to the users of smart cards. Implementing smart cards with multiple functions can be faced with obstacles that include the limited memory on-board smart cards, variations in the physical memory of different card types, requirements for high transfer rates that may be hindered by increased complexity due to the implementation of multiple functions, and quickly increasing requirements for interoperability as application providers are added to the multi-functional environment. Other limitations may also be faced.

Known techniques in this field apply rigid fixed formatting techniques in which each application of the same type must accommodate data fields needed for the other applications of the same type. Such techniques tend to occupy a substantial portion of the memory on a smart card and can be deficient in allowing for limited expansion or reconfiguration of existing applications.

In addition, data integrity in everyday use of smart cards may cause significant related obstacles to meeting smart card operational needs, for example, because of existing

demands on memory space, speed of operation, and stability of on-board data in such smart card environments.

As such, there is a need for improved techniques for user media and related applications that provide interoperability in multi-functional environments and sufficient data integrity.

SUMMARY OF THE INVENTION

In accordance with the principles of the present invention, techniques are provided for flexible implementation and operation of multiple functions for use with user media such as smart cards in order to receive the benefit of associated services. A system, which may comprise a network of equipment (*e.g.*, media readers), may be established for example by different entities. Multiple functions may be available on the system based on established configuration of user media in cooperation with software and/or hardware on the system. The system may be specific to a particular region for example to provide regional transportation services. A system may be implemented in which user media such as smart cards include data having a top-level that is organized based on categories or types of applications (*e.g.*, transportation, parking, banking, etc.). In some instances, if desired, multiple instances of the same category or type may exist at the top-level, however, it is more efficient for a single category or type to exist so that a functional hierarchy is established. Different originators in the same category or type may configure their software to use data configured below the top-level category or type and may take advantage of common data groups, data structures, or data elements that may exist because of potential overlap in the service or function that they provide to users. Thus, for example, a smart card may contain transportation at a top-level with one or more lower levels being configured for transportation software of different transportation operators (*e.g.*, operators in a region). At a lower level, multiple software, which is sometimes referred to herein as applications, of different originators may be implemented. User media (*e.g.*, a smart card) may include an application when the media includes structured data (*e.g.*, data groups, data structures, data elements, etc.) that are configured for operation with a particular executable code (*e.g.*, executable code residing on card reader equipment). Executable code may also be included on user media in combination with the data to enable an application on a smart card. Thus an application (*e.g.*, a subway transportation application) may reside on a smart card even in the absence of executable code on the smart card.

A profile may be established for an application, which may then be used to configure user media to operatively perform with the application to accomplish desired transactions and updates to stored information. A reference or standard may be the source for the profile with which different organizations can develop services or applications for use on the cards. For example, a list of data groups, related functions, data structures, and related information such as data elements may be established for applications that are to provide services in a particular field of service or industry because of the potential commonality in function or data. Multiple functions may be configured for use on the same card based on a standard list or reference while preferably also providing the capability to cross-reference known data between implemented applications.

The cards of each issuer or originator may be structured to include data groups and data elements that have been selected by the issuer or originator to the exclusion of other data groups and data elements that are also available for their application from a standard list or reference specification. Each application on user media (*e.g.* a smart card) may have an associated application identifier on the user media. An application directory may be included on the user media to assist applications to locate and/or identify data or software associated with the applications on user media. Each application or application directory may have a plurality of data groups associated with it. A data group directory may be provided on the user media for the data groups associated with an application. Each data group may include data elements that are structured together in a data structure for access by its associated application. The data group and data structure may be directly related to the functions of the application. Data groups and data structures may have been a selected subset of a list of data groups and data structures that have been defined for use with that application or applications of that type. This allows different application developers for the same type of application to implement their application differently by using a "pick and choose" technique to decide which common data groups or data structures shall be configured for their application on issued cards.

Within a network of supporting equipment, software and hardware (*e.g.*, card readers) of different organizations may be configured to operatively interact with the multi-application cards to provide services to users. Information on user media may be stored and updated at different levels of the system. Information may move from a central computer system (*e.g.*, a region computer system coordinating activities within a region) down to individual user media to write information on the media. In addition, systems and techniques illustratively described herein may be independent of the type of user media.

Data structure may be implemented without for example including tagged length value information because such information may be established within the system. Field information such as bit length, format type, character, numeric value, currency for data elements may be excluded from a data structure such that for example, only an index is included for data elements on a card. Related information regarding format type, bit length, etc. may be obtained by a system from reference information for data elements in a particular standard list when the existence of data elements are identified.

If desired, the user media may be a smart card such as a memory smart card. User media may be configured for a single application consisting of all necessary data for a particular application type, multiple applications of the same type and/or multiple applications of different types. For example, multiple transportation applications may be implemented to provide multi-application media for use in a particular region. If desired, other types of applications may also be implemented such as to provide user media that is structured for credit card applications, public transpiration applications, health care applications, etc. Alternatively, a group of applications (*e.g.* ticketing, purse) could be incorporated into a single application for use in a particular region thereby allowing improvements in transaction times, and simplifications of processing.

Features may be implemented, for example, in addition to techniques for providing interoperability, that maintain integrity of data on user media, such as through anti-tearing protection. For example, in a system that uses a data group directory for an application to identify the relative location of the desired data elements (*e.g.*, through previously specified bit formatting of data elements for applications of that type), data elements can be treated different based on classifying data into different categories such as fixed data groups and dynamic data groups. The classification may be based on read/write functionality of data elements or based on when and where user media is interfaced with system equipment. The use of different classifications allows for providing replicates or backup information for a subset of data elements of a current application for day-to-day use. Dynamic data groups can be further subcategorized into alternate, circular, or parallel data groups based, for example, on the functionality of corresponding data groups. Other dynamic data groups or combinations thereof may also be implemented. Information on the classification of data elements may be stored on user media or may be predetermined for data elements. Alternate data groups will typically only require one alternate record set for that data element. Parallel data groups will typically have multiple record sets that are stored in parallel. For parallel data groups, the application may need less than the given number of

record sets for the parallel data group to be valid at the same time. Thus, for example, a parallel data group may have five associated records wherein four of the records are valid and the fifth may be used for writing the next incoming event. Circular data groups may track a sequence of activity and may overwrite a set of records in sequence to allow for a specified number of prior sequences to be valid. By using different anti-tearing groups, the needed memory space for a current application may be reduced. In addition, the speed of operation for anti-tearing may be improved because of effective management for the validity of data records. The data groups for an application may include a valid data record list that can be read or updated with software implemented anti-tearing protection to identify valid or most recent data in view of the replicates or duplicates that exist on user media for different data groups.

BRIEF DESCRIPTION OF THE DRAWINGS

Further features and advantages of the invention can be ascertained from the following detailed description that is provided in connection with the drawing(s) described below:

FIG. 1 is a functional block diagram of software and/or hardware in accordance with one embodiment of the present invention;

FIG. 2 is a functional block diagram of card reader equipment in accordance with one embodiment of the present invention;

FIG. 3 is a functional block diagram of a regional card system in accordance with one embodiment of the present invention;

FIG. 4 is a functional block diagram of an agency system in accordance with one embodiment of the present invention;

FIG. 5 is a functional block diagram of a data group in accordance with one embodiment of the present invention;

FIG. 6 is a functional block diagram of a card configuration in accordance with one embodiment of the present invention;

FIG. 7 is a functional block diagram of a multi-application card in accordance with one embodiment of the present invention;

FIG. 8 is a functional block diagram of an application in a multi-application card in accordance with one embodiment of the present invention;

FIG. 9 is a flow chart of illustrative steps in providing implementing applications in a multi-application environment in accordance with one embodiment of the present invention;

FIG. 10 is a flow chart of illustrative steps in configuring and using cards for different applications in accordance with one embodiment of the present invention;

FIG. 11 is a flow chart of illustrative steps in configuring interoperable data groups with data integrity protection in accordance with one embodiment of the present invention;

FIG. 12 is a flow chart of illustrative steps in configuring dynamic data groups in accordance with one embodiment of the present invention;

FIG. 13 is a flow chart of illustrative steps in implementing anti-tearing in operation of applications in accordance with one embodiment of the present invention;

FIG. 14 is a flow chart of illustrative steps in implementing anti-tearing protection in accordance with one embodiment of the present invention; and

FIG. 15 is a functional block diagram illustrating valid data record lists and dynamic data groups in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

User media such as a user card (*e.g.*, a smart card) may have various circuitry on-board in the form of integrated circuits and in some cases, executable software, to provide various functionality. For example, with reference now to FIG. 1, card 100 may include input/output interface 102, memory 104, processor 106, and software 108. Card 100 may be the shape and size of generally known smart cards (*e.g.*, the shape and size of a credit card). Various techniques for producing card 100 and on-board hardware are known to those of ordinary skill in the art. Input/output interface 102 may provide for a contact or contact-less interface for communicating with the on-board memory, processor, or software of card 100. In the case of a contact-less interface an optical read/write interface may for example be provided. Processor 106 may be a microprocessor that is configured to interact with memory 104 and software 108 to provide desired functionality. Software 108 may be configured to operate on processor 106 to support a card operating system and one or more card applications. In some embodiments, card 100 may be a memory card, in which case it will not include processor 106 and software 108. Memory 104 may include RAM, ROM, EEPROM, etc. for providing storage of data, data structure, or software. In the case of processor cards, card 100 will typically include RAM, ROM, and EEPROM storage for

supporting processor card functionality. In the case of memory cards, card 100 will typically at least include EEPROM storage.

Various benefits can be provided to a cardholder through supporting applications that interact with a user's card at various distributed card reader equipment. For example, with reference now to FIG. 2, card reader equipment may include card reader 200 and client terminal 202. Card reader 200 may be configured to operatively connect with card 100, for example through contact or contact-less connections. Various card readers are presently on the market for reading and/or writing smart cards. Client terminal 202 can be a computer system (*e.g.*, a personal computer) that communicates with card reader 200 to interact with card 100. Client terminal 202 may include one or more applications 204 and other software such as an operating system. Application 204 may be applications that are each configured to provide the benefit of specific services such as transportation, health benefits, electronic wallet, etc. For example, application 204 may be a transportation application for providing various transportation such as ticketing, fare collection, etc. Interface 206 may include driver for interfacing with card reader 200. Interface 206 may also include an application program interface for providing an application interface to application 204 that allows applications 204 to interact with software, memory, or processor on-board card 100. In cases in which card 100 is a processor card, a card operating system may be used to interact with the card to execute specific card services that are available via the card operating system. The card services may for example be called through the application program interface with application 204. In the case of memory cards, interface 206, card reader 200, and application 204 are configured to work together to read and write to memory locations on-board card 100 to allow application 204 to provide desired services. The interaction between card 100 and card reader 200 is typically very short in duration (*e.g.*, hundreds of a second) to allow the cardholder to quickly interact with the system.

Card reader 200 and client terminal 202 may be one of many such equipment that are dispersed geographically and are network connected (*e.g.*, through a central database) to provide services that are associated with each different application 204 (if more than one exists) on client terminals 202 and/or its associated network.

FIG. 3 provides an illustrative arrangement for a regional transportation system which may for example incorporate equipment and networks such as that shown in FIG. 4 (discussed below). System 400 may include region central computer system 402, such that there may be multiple region central computer systems for a number of different regions. A region may have agencies (*e.g.*, transportation agencies) that have agency central computer

systems, *e.g.*, agency 1 central computer system 404, agency 2 central computer system 406, and agency 3 central computer system 408. Agency central computer systems 404, 406, and 408 may be configured to communicate with region central computer 402 to receive and transmit information so that their fare card information is updated and for example, information about fare card activity is processed and stored. Agency central computer systems may be configured to communicate and operatively interact with related subsystems that interface with user fare media, (*e.g.*, user smart cards, or other media). A subsystem may be equipment such as subway equipment (*e.g.*, fare gates, fare validation equipment, ticket office machine, remote card reader and workstation, etc.), commuter rail equipment (*e.g.*, fare validation equipment, ticket office machine, remote card reader and workstation, etc.), bus/LRV equipment (*e.g.*, fare box, on-board validation equipment, garage computer system, etc.), or other subsystem equipment. Fare media 410 may be a smart card (*e.g.*, memory card or processor card) or other media that can be used with agency subsystems and which carry data fields, structures, size, or segmentation. In operation, various data is passed through the hierarchical levels in between fare media 410 and region central computer system 402 For example, directed action data such as hot lists and changes to fare media may be communicated downward through the hierarchy towards the fare media and transactional data resulting from fare media interaction with subsystems may travel upwards through the hierarchy towards the regional system.

Regional interoperability may be permitted by applying format techniques (*e.g.*, smart card format techniques) that allow for flexibility in implementation. Preferably, the format techniques are independent from the user media technology (*e.g.*, physical card technology) that is applied in each individual case and is designed to adapt to future technological advances and future expansions in function and data that will be handled by the media (*e.g.*, user cards).

In such systems, each agency may have implemented its own transportation application, which may be reflected by the inclusion of data configuration on fare media for their application. The data configuration of each may be configured to be understood by the systems of each agency (404, 406, and 408). The smart cards include data groups, data structures, data elements, and other related data configuration for each transportation application wherein the smart cards include information identifying the data settings based on common data elements, data structure, data groups, etc., which may have been selected from a standard list.

One example of an agency system specifically for regional public transportation is illustrated in FIG. 4. Transportation services within a particular region may include a wide range of transportation types, transportation providers, ticketing techniques, ticketing validation techniques, and fare collection techniques. The number of vehicles, passengers, and geographic distribution of the passengers, vehicles, and equipment is also accounted for in the regional system. As shown, the system may include central computer 300, which may include one or more servers, for example to provide an application server, a database server, a subway server (for interacting with subway equipment), bus/land roaming vehicles ("LRV") equipment, commuter rail equipment, etc.

The regional transportation system may include bus/LRV equipment 302, retail sales equipment 304, subway equipment 306, commuter rail equipment 308, customer service equipment 310, ticket office equipment 312, and central encoding equipment 314. As shown, various (wired or wireless) communications network techniques such as LAN, WAN, Ethernet TCP/IP, Internet, fiber, T-1, PP, ISDN, switches, or routers, and variation thereof may be configured to establish a network for supporting a transportation application within the system. Central encoding equipment 314 may include a smart card initialization machine and if desired, photo ID equipment. The smart card initialization machine may be configured to initialize individual cards that are issued with appropriate security and transportation data. The photo ID equipment may be configured to receive, store, or generate photo identifications for individual cards.

Ticket office equipment 312 may, for example, be one or more ticket office machines (*e.g.*, a stand-alone ticket office machine) for providing card services. Customer service equipment 310 may for example include one or more trucks that carry fare validation machines for providing card services. Communications with the network such as with central computer 300 may be established when customer service equipment 310 (*e.g.*, trucks) are at a transportation system service facility such as a revenue facility. Retail sales equipment may include a retail sales terminal and communications equipment (*e.g.*, a telephone) for communicating with the network. Bus/LRV equipment 302 may include garage computer system 316 and bus/LRV 318. Garage computer system 316 may include computer equipment that is configured to communicate with the network and communicate with fare validation equipment on-board a bus or LRV (*e.g.*, through an RF connections). Bus/LRV 318 may for example be a bus or other vehicle that has a fare box and a flash memory module for reading and updating user cards. Subway equipment 306 and commuter rail equipment 308 may each include fare validation machines, station controllers, station information

centers with ticket office machines and network capability to support remote smart card readers with cradle and workstations (*e.g.*, a user's home computer equipment) to provide transportation card services to users. Gates are also typically included as part of subway equipment 306. Central computer 300 may for example include an application server and a database server that interact with network nodes such as subway equipment 306 or bus/LRV equipment 302 to provide software downloads, application functionality, or a central card information database, which is updated based on network interaction with cards. As such, the system may be configured to permit a user to ride different transportation that may be offered by a single agencies or if desired by different agencies (*e.g.*, commuter rail, metropolitan transit, ferries, etc.) for possibly different fare structures (*e.g.*, one way, weekly, point-to-point, unlimited ride, etc.).

The techniques illustratively described are adaptable to meet many different needs. As such, formatting techniques, user media, and supporting systems can be established that allow for standardized hardware and software platforms for media products such as fare cards that can be licensed by neutral or non-vendor parties and available to operators as well as contractors for the purpose of providing equipment and services for a particular industry that has implemented such user media services (*e.g.*, smart card fare media in the transportation industry), while applying compatible solutions in order to allow for interoperability and open competition among providers. Such techniques may be specific to smart card applications such as ISO 14443 type A&B smart cards and dual interface smart cards, but are preferably media independent.

A flexible framework can be established that for example can address the current requirements of a particular industry (*e.g.*, the transit industry), which allows a "pick and choose" type functionality for establishing user media (*e.g.*, smart cards) for each service provider. For example, in the context of regional transportation services, this type of design provides a flexible framework that allows for configuration of regional fare media products in combination with "local" fare media products while maintaining data security, integrity, privacy and flexibility to allow for future expansion. The framework, for example, establishes data groups, data structures, and data elements that are grouped or associated based on the desired function of the data, combines mandatory and optional data elements for a data profile that corresponds to functional requirements of desired application features, is adaptable to memory size that is available for a corresponding application on user media such as a smart card specifically issued for one service, as well as on a third party card such as a credit or debit cards, and establishes an open architecture to allow integration of new features.

Other aspects may involve specifying a list of functional data groups, which have specific elements assigned to them, implementing a list of mandatory data elements that may be needed for interoperable use, establishing techniques for implementing an application on different media types with different memory size and organization, establishing options for dedicated user-media applications, and establishing the combination of different user-media applications (*e.g.*, combining transit applications and credit card application on a card).

A modular approach is applied, in which "building blocks" are provided that can be used by each application issuer to configure the overall "structure" each issuer would like to implement on their user media (*e.g.*, cards). Data types are selected for use on a user media to support required functionalities. Different data elements will, thus, be part of different functions, while desired functions are used to determine which one of potentially many different data elements will be on-board a card for storing data content or values for those data elements. The data organization of user media can for example be structured as illustratively shown in FIG. 5. User media may be structured into one or more data groups 502. Data group 502 may be structured to include one or more data structures 504 and 506, which can be further structured into one or more data elements 508 and 510. The hierarchy allows for the association of a function or application with data group 502 and the association of incremental levels of functional detail. Flexibility may be established by providing "super sets" of data fields or functions which are adaptable through subsets for desired functionality or application. For example, in a particular field of industry or services, a standard list of data groups, associated data structures, and associated data elements can be established that can be structured into subsets by originators of applications in that field of industry or services to meet their needs and to operatively implement for their individual applications in that field of industry or services. Data elements or structures that are not selected are entirely left out of an application except for example by including bit or flag indicating its absence (*e.g.*, the bit size of an excluded data element will not contribute the amount of memory occupied by the application).

Such modular building blocks may be applied for example as illustratively shown in FIG. 6 to smart cards. Memory media on smart cards may be structured to include card base data 602, card application layer 604, data group directory 606, data group layer 608, and data structure layer 610. Data group layer 608 may include a plurality of data groups 612. Each data group 612 preferably corresponds to a function. (*e.g.*, stored value, ticketing, and loyalty program by a specific entity, etc.). These data groups 612 are preferably subdivided into data structures 614 in data structure layer 610. Definitions of data

groups 612 and data structure 614 may be logically driven by functional requirements and card limitations. For example, memory-type smart cards have a fixed layout, and record size, and, therefore, with an appropriate data structure definition, a proposed memory structure may fit onto all memory cards. Microprocessor-type smart cards, however, provide for different structures, and correspondingly, allow files to be configured in almost any arbitrary size.

Data structures 614 are preferably further subdivided into data elements. The data element can be considered the lowest structural "unit" of measure on a card. Based on these functional data groups, what is needed may be "built" onto a card in an efficient manner based on a specific originator's (*e.g.*, transportation agency's) needs, while also allowing interoperability. A benefit of this design approach is that there is more flexibility for customization. Furthermore, on memory-type smart cards, variations in record size from one memory-type to another will not affect the card format. In addition, data elements can be varied without a tedious reorganization of element order and element definitions. Elements can also be added and subtracted with ease. A change in data group composition can be driven by the evolving needs of an application originator (*e.g.*, a transit agency), which results in a high degree of likelihood that data group and data element definitions would have to be changed.

Data structures 614 and related formatting can be configured to offer a "map" (via a bitmap included at the beginning of a data structure) that indicates the data elements that are currently applicable and in use in that data group. The bitmap can preferably identify which data elements from a list of data elements were selected in configuring the current data structure (*e.g.*, the current sequence of bits). The bitmap can also reflect the size of the data structure in that selected data elements, which may have defined bit lengths, contribute to the size of the data structure, while data elements that are not selected preferably do not contribute to the bit length of the data structure. Thus, new data elements can be specified and selectively added to the end of a data group or data structure, as needed. By using the bitmap, different systems can read slightly different configurations and implementations of the card format. The addition of new functions and data elements can be added without having to redesign the whole layout or adding new files.

In organization, card base data 602 may be an application directory that includes information identifying or pointing to the location of data associated with an application or an application category or type (*e.g.*, an application category or type that holds one more applications in that category) in the application directory. Card base data 602 may

include application identifiers or information from which a card application or category is identified. Card base data 602 may, if desired, identify locations on a card that values for different application identities for applications that have associated data on-board the card. For example, certain groups of bits may include information identifying a particular credit card company. Data group directory 606 may include information identifying different data groups 612 that are associated with that particular data group directory 606 and may contain related location or pointer information. Location and/or content information may also be associated with each data group in connection with associated data structures 614. Data group directory 606 may be associated with a corresponding application 604 or applications in the same application category or type.

Having a "flexible" structure partition has technical benefits as well as those of convenience. For example, currently, operations on common MIFARE type cards, which are known to those of ordinary skill in the art, are performed with data transfers of 16 byte blocks. This also corresponds to the record sizes on many other memory-type smart cards. Each operation performed can typically only deal with 16 bytes at a time. However, in the case where the rate of operation increases, for example, at a rate of 64 bytes per operation, a record size on a card of 16 bytes would require 4 operations to perform the same task that could be performed in 1 operation with the "flexible" structure partition. With the fixed 16 byte record size, four times the necessary transaction time would effectively be required.

In the future case where there is a card with 32 byte records, as opposed to 16 byte records, if a format with 16 byte records (or any other arbitrary fixed size) is to be "transferred" to a 32 byte record format, it would not be a trivial matter to map two 16 byte records onto one 32 byte record. The "quick and easy" solution would be to map one 16 byte record, onto one 32 byte record, which would result in an excess of 16 bytes of each record that is wasted. The techniques and systems of the present invention preclude the need for this. This approach assumes that an application may be used on different card types that may have different memory sizes and memory structures and accommodates for this possibility.

In the context of the transit industry, certain functions are typically used in implementing ticketing medium. A smart card may be configured to contain predefined data for a specific ticket type including time and distance parameters. This ticket can be used as single, round trip or multiple ride ticket or pass (*e.g.*, weekly, monthly passes) based on its time and distance parameters. Validation may not be required when the smart card is used as a pass. A usage profile, based on check-in and check-out events, can be stored on the pass to recognize invalid usage. A validation or partial validation of the smart card is required prior

to or during the journey in the case of single, roundtrip, or multiple-ride ticket types. Automated access to a "Paid Area" of a transit system will be given by check-in/check-out systems. Time and location can be used by an originator's implementation of a transit application to determine the tariff calculation for a current rider. Check-in and check out "stamps" may be generated and stored on the smart card.

Customers that are registered can receive benefits for their tariff calculation or payment methods. Otherwise, customers may use their fare medium without on-board data containing personally associated data for that customer. User specific information stored on the card can be utilized for the tariff calculation to determine an adequate ticket type either before the trip, (via ticket vending machine or ticket office machine) or automatically at check in/check out use. The smart card may also function as an ID-card for access control for transit agency staff. For example, image information may be stored on smart cards that can be accessed to identify the current cardholder.

Account-based smart cards may be implemented, where for example, payment can be made via a user-associated account, which may for example be only available to registered customers. In account-based smart card structures, a user can pay immediately or the user's account will be charged for predefined tickets before the trip. In check-in/check-out systems, the customer's account can be charged based upon usage transaction data (*e.g.*, time or distance). A calculation can also be handled by the account provider, which may conduct processing to suggest or select the best fare under the circumstances.

If desired, a preloaded value can be stored either in a purse on the smart cards that can only be used by public transportation, or stored in a purse that could be used by additional card applications, (*e.g.*, park and ride). The purse may be considered a "closed purse" with defined performances and processing protocols, as opposed to an "open purse" such as a public "electronic purse." The preloaded value can be used to purchase a predefined ticket type (*e.g.*, before a trip), or as a pass that can be used through a system with check-in/check-out devices. This is based on the tariff system and uses deducted values from the purse. The value of the preloaded purse can be represented in any unit of currency. The purse can of course, be added to, when initial value is exhausted. The "value add" could be paid immediately (*e.g.*, on the spot by the user) or by charging the user's account.

The ticketing card application may be loaded onto a smart card issued by a bank. For example, data structures, formatting, and content can be loaded onto the smart card of the bank that can be operatively used with a transit application. This smart card may already have an existing e-purse configured on-board. Values could for example be deducted

directly from the e-purse based on purchases of predefined ticket types. E-purse values could also be altered by processing with check-in/check-out systems or by adding value to a stored value purse or ticketing application. Various applications (*e.g.*, executable software, application data, or the combination) and payment processes can co-exist on a single card.

With the techniques and systems of the present invention that are illustratively described herein, an interoperable smart card can be provided for example in the context of the regional transit industry, that can be accepted by terminals of different transportation agencies that have different tariff systems. Different ticket types can be predefined on the system and stored on individual smart cards to be valid throughout the interoperational transit regions (*e.g.*, a monthly ticket that is valid for the month for subways, ferries, and commuter rail). With such tickets types, services of different transit originators can be combined to benefit the user with packaging of services from multiple transit agencies, while in each case, the smart card may for example be structured to include a separate group directory and associated data structures for different applications of each agency that are operable with the cards. Stored check-in/check-out events can be recognized and processed for inter-agency transfers. For example, a check-in/check-out event from a commuter rail can be recognized by terminal equipment on-board bus transportation of a metropolitan transit provider to identify a free transfer. The systems may be configured to accept the discount condition of a personalized card throughout the transit region, even if there are different tariff schemes. For example, a user may have registered with one of a plurality of transit providers in a region to receive a discount on fares, which can be extended by the system to other transit providers in that region based on the user's region based on the user's registration. Another aspect, provides for a closed stored value purse for applications in that industry or field of services (*e.g.*, providing different types of the same service) is accepted by the ticketing application of each provider for payment. Thus, a purse, that may have associated values stored in connection with multiple transit applications, may be closed for use to a group of applications (*e.g.*, transit applications in region), such that a user can apply value from the purse to receive benefits of the same type or benefits from providers in the same field of services of industry (*e.g.*, a user can decide on how he wants spend his transportation budget for the month). Another aspect may provide for additional ticket types stored on the smart card that are valid in a specific transit region or agency.

Further aspects, for example, may provide for additional ticket types stored on the smart card that are valid only to a specific transit agency or may provide for other agency-specific functions such as loyalty or other programs that are valid only to a specific transit

agency. For transactions that are related to multiple agencies, a clearing procedure may be established. Required data, *e.g.*, identity profiles, and user profiles, etc. are to be included in a card application in order to provide required transaction data for clearing procedures. For example, transactions with one agency may be related to data of an application of another agency on-board a card. In such cases, the transaction may be cleared with the other applications.

FIG. 7 illustrates in functional diagram form, multi-application card 702 that provides one example of applications or categories of applications, *e.g.*, public transport 704, being integrated onto a third party card. Public transport 704 may also be fully functional as a standalone application (or a standalone application category containing one or more applications) on a card. Multi-application card 702 may be a smart card or may be some other media. Other types of applications, other than transit, may also be configured for multi-application cards of a third party and if desired, further configured to be individually fully functional on card. Card base data 706 may identify and/or point to on-board card applications or categories of card applications, which may only contain data or contain data and executable code. Card applications on card 702 may include credit card 708, open e-purse 710, public transport 704, car parking 712, or other 714. a cardholder may use card 702 via card readers to receive the benefit of services associated with each category.

FIG. 8 illustrates the layout of different functional data groups in a card application, such as public transport application 704 of FIG. 7. Card application 802 may include data group directory that may include data group identifiers and/or location information such as data groups 806. Data group 806 may for example include a data group for the identity of the application, a data group for user data, a data group for tickets, a data group for a loaded value purse, and potentially other data groups. Thus, as shown in FIG. 8, data groups 806 and functions are referenced by directory 804.

Application 802 preferably contains a set of data structures, such as data groups 806 that define information directly related to a function stored on the card. Each application on a card may be structured on the same basis. A data group preferably contains data structures, that may for example be composed of header information, and data elements that contain the actual data details. All individual data of each card application are preferably structured together in separate functional data groups 806. Access to data groups 806 will be processed through data group references, specifically in this case, data group directory 804. Data group directory 804 may be structured as an open list and may include links to all different data groups 806 for that application 802 or applications in that category of

applications. A data group identifier may be established in association with a corresponding data group (*e.g.*, each data group is assigned a data group identifier). A data group may be absolutely referenced by its data group identifier. The identifiers may be used to absolutely describe a specific function in a card application. Depending on the physics of the card, a formal reference to an individual group may be assigned (*e.g.*, data structure(sections, blocks, files)). Preferably, data structures and their data elements are grouped together in data groups 806 based on their functional structure. If desired, functionally identical data groups could be specified multiple times in an application. In addition, different data groups 806 could contain the same data structure and data elements. Thus, mutual exclusivity of data elements or data structure may not be required at the data structure layer. If access rights are desired, a defined access right may be assigned to all data elements of a data group. Table 1 and Table 2 below schematically illustrate a data group directory and data group.

Table 1

Data Group Directory	
Data Group Identifier	Reference (Format based on card physics)
1	Reference to data group 1
2	Reference to data group 2
..	Reference to data group n

Table 2

Data structure x	
	Data element 1
	Data element 2
	Data element..
Data structure y	
	Data element 1
	Data element 2
	Data element..

Data groups 806 and the data structures assigned to them are preferably defined based on functional requirements. The physical specification of card types may define data storage restrictions with respect to data size as well as the memory structure of each individual card (sector, block, data structure). In addition, the physical card requirements will specify the physical location and the procedures by which access to the data on the media can be achieved. The physical card specifications and restrictions may be included in the definitions for a card application in order to optimize the system performance and the memory size requirements. An application in a distinct environment should preferably be implemented based on the physical card requirements/restrictions. In implementation, card operating systems, drivers, or card terminal software provides for appropriate translations in executing card applications, for example, by referencing the logical data structure to the physical implementation, which is dependent on the card type.

Data groups, as noted above, may be configured in a number of different ways. For example, each data group may be configured to comprise data structures. Each data structure may preferably include a header and a number of data elements. Data elements may be assigned to different data groups based on particular criteria. Assignments of data elements to specific data groups may be based on group function, read/write criteria assigned to the data group, which may often be required to be the same for all data elements inside the group), mandatory or optional category of a data group or other criteria.

A data group may be identified based upon a data group identifier (*e.g.*, a unique data group identifier), which can be a special data element of each data group structure. The identifiers can specify a unique functionality inside a card application.

In implementation, there may be some data groups that are mandatory requirements of an application, and are designated as such. An "identification" data group, which can describe the application, is preferably one of these mandatory data groups. In order to handle multiple applications (*e.g.*, multiple applications in the same category or type of applications) of the described structure by using the same access mechanisms, the data group identifiers of mandatory groups may be globally specified. This requirement may be established to be valid in all environments (*e.g.*, in card terminals and PC-cards). In addition there may be certain unique mandatory data groups, which may only be specified once in an application. Functions added to an active application can be inserted as new data groups are added into the data group reference.

The data elements of a data group are preferably combined in data structures. The sequence of individual data elements, their function, and data format (type, data range,

size) in a data structure can be specified by a structure definition. Such information may be established in a standard reference specification for data elements in a particular category of applications and/or established as part of the data structure (*e.g.*, in a header to identify selected elements and values related to a selected element such as size, value, etc.). If a standard specification is established, values related to a data element can be automatically identified once it is recognized that the particular element is part of a current application. A structure-identifier (*e.g.*, a unique structure identifier) may preferably be positioned at the beginning of each structure. The structure length which may follow directly after the structure identifier can specify the size of the complete structure. This can enable efficient bypass of data elements within a data structures as well as the addition of individual structures that are added for future features. It can also enable the application to directly jump to the next structure.

The existing data elements can be referenced in a data element directory structured in a bitmap. As such, it is possible to delete or add optional data elements by configuration of the bitmap based on the individual implementations. If data elements are added, they can be placed at the end of the data structure, and its addition may be reflected in the bitmap. The maximum number of data elements included in a data structure may be specified by and limited to the value of the structure length or the data structure directory.

Data Structure						
Structure ID	Length	Bitmap	Data Element 1	Data Element 2	.....	Data Element n

Element	Bits	Value range	unit
StructureID	6	1-64	
Length of bitmap	2	0-3	1 to 4 Byte
Length of structure	8	3-255	Byte (entire structure)
Bitmap	8,16,24,32		
Data Element 1	1 — 255*8		
Data_Element 2			
.....			

Table 3: Table Illustrating Data Structure Detail

If structures are used exclusively in a specific data group, a unique structure ID is defined by the combination of a data group identifier and a structure identifier. In another aspect, multiple equal data structures could be assigned to the same data group. In order to use the same structure in different data groups, the structure itself should be defined accordingly. Based on this, a formatting structure may be applied in which part of the available range of structure identifier are assigned to describe data structures independent from the data group and another part of the available identifiers can be re-used with different data group identifiers.

Profiling may be applied to applications or application requirements to assign functions and associated data elements (*e.g.*, from a template) to a particular application and to structure cards based on the assigned functions and data elements. Local functional requirements and card memory space required could vary in an interoperable environment. Based on this there may be different profiles that should be defined for a particular application (*e.g.*, an application in a particular category) to accommodate the variations. All smart card terminals and related equipment, however, may be required to process the minimal functions of different card profiles in an interoperable environment. This can for example be supported by a smart card application carrying its own profile, which can be provided using data group identifiers, structure identifiers, structure lengths, and data element directories for the structures.

If desired, additional data groups may be added and/or new data structures may be added to data groups. New data groups and/or structures may be defined, or additional/new data groups may be defined out of existing structures (*e.g.*, created from specified data elements or data structures). In one approach, new data elements may be added to the end of an existing data structure.

Interoperability may be applied based on standardization in formatting techniques and application of such techniques within a network of card terminal equipment. For example, illustrative steps are shown in FIG. 9 for implementing multiple applications in a multi-application system environment. At step 902, equipment such as the equipment of FIG. 3 may be configured to execute applications of different originators (*e.g.*, application developers, service vendor associated with an application, card issuer, organization providing the benefit of category of services, etc.). Applications may include individual public transportation applications, individual financial applications, or other applications.

At step 904, the equipment may be configured to be in a network to share data between equipment of different originators and to for example provide storage for user media

information such as a central storage for updating related user information across the equipment of different originators. At step 906, networked equipment may interact with user media such as smart cards. Step 906 may for example include processes of step 908. At step 908, an application related to a current interaction with card reader equipment may be identified on-board user media. Data groups associated with the identified application may also be identified and/or located. The data groups may have been selected for use in the identified applications from a standard list of data groups and related definitions that the originator considered in implementing their application. The data groups may include data elements which may have been similarly selected. As part of the interaction, the equipment may read or write to appropriate physical memory (*e.g.*, the identified application and/or supporting software may read or write the appropriate physical memory location independent of the current media type of user media).

At step 910, information on the network may be updated in connection with the interaction (*e.g.*, a central storage or originator storage may be updated based on the interaction). Step 910 may include step 912 for clearing updates resulting from the interaction through other applications with which the user media is structured. Thus, interaction that may be related to data on other applications may be addressed through a clearing procedure by which memory on-board user media or external user-media may be appropriately updated.

A standard or reference for functions, sub-functions, and related data groups and data structures may establish flexibility in the implementation of applications while providing interoperability independent of issuer, originator, equipment vendor, equipment operator, or card type such that defined functions, sub-functions, data groups, and data structures can be applied to cross-reference or reuse between applications. Illustrative steps for providing techniques for multi-application interoperability are provided in FIG. 10. At step 1002, a profile may be established for an application to be used in the multi-application environment. A profile may be established for each application. A profile may be established by for example the originator of an application by identifying specifics from a standard or reference for functions, sub-functions, data groups, data structures, and other factors that can contribute to support the particular implementation of that type of application by the originator.

At step 1004, a profile may be applied to cards that are to be used by cardholders by configuring the cards based on the profile. Other user media may also be configured and used in illustrative steps of FIG. 10. Step 1004 may include steps 1006, 1008,

1010, and 1012. To apply the profile, an application identifier may be established for an application and configured on the cards for identification by card reader equipment. A data group directory may be configured in connection with an application or application identifier (or multiple applications/application identifiers in the an application category), which may include identification and/or location (*e.g.*, pointer information) for data groups that were profiled for that application. The cards may be further structured to include data groups that are identified in the data group directory. Data groups may correspond to functions within an application which may been added to the profile on that basis. In addition, data structures, which may for example include a header and related data elements, may be configured for the cards. The data structures may have been selected to be part of the profile based on corresponding sub-functionality. There may also be mandatory data groups or data elements which may have to be established as part of this procedure. The data structures and the data groups may specifically identify which data structures and data groups were included as part of a profile by an application provider to be include on the system in connection with their application.

At step 1014, the configured cards may be used at card reader equipment in connection with different services (*e.g.*, in a particular region) to receive the benefits associated with individual applications operating in the multi-application system and to store or update data associated with the data groups or data structures on-board the user cards and/or on the equipment of service providers.

For interoperable use of an application between different providers, it is necessary to have the option to customize the data groups specifically for it. Interoperability could require that specific options would be used in the specified main area only. The use of the same feature for other operators would be limited to the minimum required data groups. In order to include additional or new user specific data from individual users into the application, it may for example be necessary to store the individual user profiles in separate data groups.

There may be particular data elements that are structured on user cards that can support interoperability or may be required for establishing interoperability. An application identifier may provide direct access to an application via an application directory or a corresponding card command with the declaration of an application identifier. These application identifier declarations, which may be unique identifiers, may be related to the card type and/or the card issuer (a card standard, a card registration). As such, the same application could be assigned different card application identifiers. For the processing of

logical card data and transaction data in supporting systems, a different application identifier may be implemented that may be a unique application identifier within the described application layout. This application identifier may specify the "functionality" of the application and as noted above is part of the data group application identity. This Application identifier is typically not related to the card type.

The version of the application describes the specific profile and the different kinds of functionalities of the application. The application version may be needed to support interoperability. Application versions should be unique relating to an application identifier only. This means the version should be independent between different application manufacturers. An application issuer region may be needed for interoperability in some types of applications (*e.g.*, region public transportation applications). As long as application manufacturers cannot be clearly distinguished by means of the areas of use of the card structure, a region mark may lead to a demarcation, and specifies the region to which the application issuer belongs.

The application issuer may also represent data that may be needed for interoperability. An application may be profiled and administered by an application issuer. Simultaneous to the profiling the issuer may have the duty of overseeing key administration for the application. The application issuer may be the same as the card manufacturer, and it may be identical to the only acceptance provider as well. For application issuers, an identifier (*e.g.*, a unique identifier) is allocated. This may for example be unique within the application that operates with this layout. Currency in the application may be also of importance for interoperability. All amounts, including the implemented stored value purse, may use the referenced currency (*e.g.*, may only use that currency).

The application language may also be needed for interoperability. This is a basic language marker of the application within its area of use. A personalized card may have a user-referring language marker in the user identity that deviates from the language marker of the application. Another component for interoperability may be an application serial number. Every installation of an application on a card may have its own identity for which the utilization, outputs, and payments are at issue. The lack of ambiguity of this identity may be supported by a serial number. An application can be installed on a card before a personalization takes place. As a result, the user identity may not provide a contribution to the serial number. Also, in the case that a third party card manufacturer has the authorization to transfer an active ticketing application from an expired card to a new card, the unique identity of the card (card serial number) may not provide a contribution to

this identity either. An application serial number, unique with an application issuer, may be provided. If the transfer of an application to a new card is not planned, the card serial number could be used as an application serial number. However, this always leads to a new identity and must be considered by the assignment to the supporting systems. Key and signature processing must take care that the transfer of an application to a different card as a simple "copy" does not lead to a valid application.

The physical card type may also be important for interoperability. For cards of the same physical card type, different configurations of the application can have consequences from the perspective of card readers and related equipment. Even if a card serial number is provided, it may not provide information about such variations. The installation time of the card application may also be important for interoperability. The installation time records the time that the card application was installed.

Data group application condition information may also be important for interoperability. The start of validity of an application may be structured to limit the date from which the application is valid (*e.g.*, version 2.0 may be loaded and set to be made valid after a certain date). An end validity of application may also be implemented. An application may have a defined end validity date when issued. In that case, an update of this element must be made in the system as a special transaction if necessary. The status of the application may also be needed for interoperability. The actual condition of the whole application may for example be assigned the following values: 0— installed (not released), 1 — active, 2— n locked (include reason for lock). Detailed information of the respective reason, source and time of a change of the condition may be shown in a data group having the function of history of common application events. Personalization may be a further category of data for interoperability, which may for example specify information on whether the current user is an anonymous or known user. A data group for the history of common application events may also be necessary for interoperability. These events may change the condition of the data group, "application condition." Application events are events that refer to the whole application. If for example, an application is locked, all application functions are unusable. Note, however, that locking a stored value purse within the application does not necessarily lead to the lock of a valid time card, which is not considered to be an application event.

Application event types may also be specified. Values for the event types describe a type of event or an operation. Values may for example indicate that an event or operation is installed, personalized, unlocked, or locked. Other related data that may be

included for interoperability may be application even time, which provide a timestamp for when an event or transaction occurred, a transaction reference identifier, the operator terminal responsible for the event, the card terminal or device involved in the transaction/event, or other data that may support interoperability.

The techniques, including the methods and systems, illustratively described herein are sometimes primarily discussed in the context of cards, smart cards, or specifically memory smart cards. However, the techniques may also be applied to other forms or types of users media.

It will be understood by one of ordinary skill in the art that software keys may be implemented in connection with individual cards or applications. As such, card reader equipment will typically use corresponding key information to access applications or to access categories or types of applications (*e.g.*, a key for transportation). In some instances, knowledge of a particular identification for a category or application may simply be used. Security may also be applied on lower hierarchy. For example, in MIFARE cards, access rights may be set differently from sector to sector so that data having the same access rights should be in the same sector(s). Keys may also be pre-assigned to individual application providers. For example, if there are two different transportation applications different keys or security setting may be established for the ability to access each. Thus, card reader equipment may apply required security processing for each transportation application to be able to access data in each. Security techniques such as those illustratively known to those of ordinary skill in the art such as security techniques implemented on smart cards may be implemented in connection with the techniques illustratively described herein.

Thus, smart card systems and method can be implemented in which originators of related services may refer to a standard list of data elements that for example specifies length and formatting of data elements so that each originator may structure their applications (*e.g.*, structure their data) differently and can identify which data elements were included for an application so that systems for that type of application can operate with the predetermined knowledge of a standard list to quickly identify, locate, and read data elements where the size of the data elements may vary from application to application of the same type. In addition, applications of different originators in the same field of service may be able to read and execute the data of other originators in the same field of service based on the standard list and the use of indexing techniques for data elements, which can reflect the current size and location of elements. Multiple applications may be configured within a card

category (*e.g.*, public transport, car parking, etc.) that can be configured to be operable within the systems of different originators of application in that category.

The systems and methods illustratively described herein can be considered to be fare media independent. However, such systems and methods may in some instances be particularly suited for smart cards such as memory smart cards. Using these systems and methods, a service provider can provide a secure card that can implement a desired functionality. For example, logical profiling may be applied using a profiling software or other means to configure cards for a service provider on the basis of a general specification or index. Logical profiling can be performed for a number of different media such as for ISO 14443 A&B cards, which may result in different layouts, for example, because of the benefits of B-type cards.

In the techniques illustratively described herein, where a service provider can pick and choose data fields that will occupy memory space on user media, a configuration process may be implemented in which requirements are identified by the service provider, the card application structure is determined, data groups and data elements for that type of service are identified (*e.g.*, in a service class specification), an appropriate profile of the data groups and data elements is identified, and the data layout of the selected profile is established based, for example, on the memory structure of the type of user media involved.

The identification of requirements by the service provider or originator in the transit context may for example involve assessing system requirements (*e.g.*, clearing requirements, anti-tear strategies, transaction speed requirements, and other system protocols), assessing fare structure requirements, and assessing application requirements (*e.g.*, the number of applications other than transit that may exist). The determination of a card application structure may involve building a card application structure such as a logical model for an application directory and an application tree. Identifying data groups and data elements may involve implementing a logical model for application data groups and data elements. Profiling may involve converting a logical data model to physical data model for a particular card structure. Establishing a data layout may involve establishing data layouts for a number of different media using the same profile, such as to establish data layouts for MIFARE cards, processor smart cards, and magnetic tickets on the basis of the same profile identified for an application of a service provider.

Different functional requirements may be faced in implementing user media. For example, in the context of transit applications, requirements of two different product contracts (*e.g.*, autoloads, user-purchased, etc.), three different actual products (*e.g.*, tickets,

seasonal, trip based, regional/local use), requirement for storage of five usage events, and a stored value purse may be required. Based on these data groups may be selected from an application related set of data groups. In the given example, the following data groups and associated occurrences of each data group may be selected, application identity(one occurrence), application condition (one occurrence), user identity (one occurrence), different product contracts (two occurrences), different actual products (tickets) (three occurrences), number of usage events/history (five occurrences), stored value purse configuration (one occurrence), and stored value purse (one occurrence).

Table 4 illustratively provides data field configuration for the selected data groups and identifies some data groups that were not selected based when the present profile was established. Information on whether a data group is mandatory or optional is also provided for illustrative purposes.

Data group name	Selected items	Data-group-ID	Number of records	Bytes needed	Write access	Mandatory! Optional
Application identity	**x**	**1**	**1**	**12**	**once**	**M**
Application condition	**x**	**2**	**1**	**10**	**multiple**	**M**
History common application events		3	0	-	multiple	O
user identity	**x**	**4**	**1**	**10**	**multiple**	**MI**
product contracts	**x**	**5**	**2**	**26**	**multiple**	**M2**
actual products	**x**	**6**	**3**	**69**	**multiple**	**M3**
usage	**x**	**7**	**5**	**75**	**multiple**	**M4**
Stored value purse configuration	**x**	**8**	**1**	**6**	**multiple**	**M5**
Stored value purse	**x**	**9**	**1**	**6**	**multiple**	**M5**
Purse event history		10	0		multiple	O5
Summary				**214**		

M	Mandatory
Mn	Mandatory for specific conditions
MI	For personal cards
M2	For identity function for pre-determined products with user related parameters
M3	Always when pre-determined products are stored
M4	Especially when check in / check out leads to fare creation
M5	If value purse is implemented
O	Optional

Table 4

A corresponding example of a layout for the above functional requirements is detailed in Table 5 below. This layout may have been established based on Table 4 for

MIFARE type media. It is understood that although other types of fare media are similar and that modifications could be appropriate. In the given examples, an illustrative block byte size of 16 bytes is being used in which 2 of the bytes are reserved for a cycle redundancy check (CRC). Different techniques for CRC are known to those of ordinary skill in the art. If desired, smart cards or user media may be implemented without CRC.

Sector	Block	Bytes in Block	Used by data group:
Appl.-Sector 0	Block 0	0— 13	Data group directory
	Block 1	0—5	Data group directory
	Block 1	6— 13	Application identity
	Block 2	0—1	Application Identity
	Block 2	2—11	User identity
Appl.-Sector 1	Block 0	0— 12	Product contracts
	Block 1	0— 12	Product contracts
	Block 2	0—5	Stored value purse configuration
Appl.-Sector 2	Block 0	0—9	Application condition
	Block 1	0—5	Stored value purse
	Block 2	0-13	Actual products
Appl.-Sector 3	Block 0	0— 13	Actual products
	Block 1	0—13	Actual products
	Block 2	0— 13	Actual products
Appl.-Sector 4	Block 0	0— 12	Actual products
	Block 1	0— 13	Usage
	Block 2	0— 13	Usage
Appl.-Sector 5	Block 0	0— 13	Usage
	Block 1	0—13	Usage
	Block 2 0	0—13	Usage
Appl. -Sector 6	Block 0 0	0—4	Usage

Table 5: Sample Layout for the Above Specification Without Anti-tearing

In the present example, an application directory similar to that of Mifare cards may be used that may require one card sector for the application directory, and a key system for access purposes may also be used. Besides the example application above, other applications for different services could be addressed by the card application directory as long as card memory size is sufficient for this purpose. As is known by those of ordinary skill in the art, the layout of the directory of user media such as smart cards is typically defined by the card type on which a data layout is implemented. The following data configurations provide examples of card related memory space based on an illustrative transit application having the selected data groups identified above in Table 4. Typically, in user media, the data group directory of an on-board application provides links to all data groups that are configured for that application. For interoperable the use of cards, card readers may be able to determine the correct location of data groups, for example, by providing within the configuration structures a link from a card application directory to an on-board application,

which in turn may be provide information or links to the data group directory.

For the example application and related data groups, there may for example be a directory that occupies 20 total bytes that may include 1 header, 8 entries, and 1 trailer that is each composed of 2 bytes. The directory may for example be assumed to start at Sector 0/ Block 0. Table 6 below illustrates the directory that would result from the sample specification as described herein. Entries are sorted by memory index (sector/block/offset). Examples of data identification numbers associated with corresponding data groups are also included in Table 6.

Data group directory configuration header					
Value Range Datagroups	Value Range Sectors	Value Range Blocks	Value Range StartOffset	Spare	
4	4	2	4	2	
Data group directory entries					
DatagroupID	ApplSectorNo	BlockNo	StartOffset	Spare	
1	0	1	6	0	Application identity
4	0	2	2	0	User identity
5	1	0	0	0	Product contracts
8	1	2	0	0	Stored value purse configuration
2	2	0	0	0	Application condition
9	2	1	0	0	Stored value purse
6	2	2	0	0	Actual products
7	4	1	0	0	Usage
Data group directory trailer					
0	6	0	15	0	End of List

Table 6

The use of smart cards or other user media in applications such as transit application is often marked by the characteristics of user driven handling. In such applications, the smart cards are not under the full control of a read/write device during the processing of a transaction. While pure read transactions are relatively uncritical, write transactions are more critical to the normal operation of an application. In conventional memory card technology, the smallest writeable data unit is a block of 16 bytes and the memory is structured into sectors of 64 bytes where each sector is divided into 4 data blocks of 16 bytes each. The last block of each sector is reserved to hold access keys and determine the access conditions to the data in the sector. Therefore, in conventional memory cards, only 48 bytes of data are available for applications in each sector. Because of the functionality and complexity of applications such as transit applications, it is often necessary to use more than one sector on a memory card or other user media to store the relevant application data.

The techniques illustratively described herein should not be considered to be limited to the configuration of conventional memory cards. Although in some instances, these techniques may be particularly suited for such memory cards. For clarity and convenience, the examples provided herein are provided sometimes primarily in the context of memory smart cards, which are sometimes referred to as memory cards. The techniques may also be applied to other media even though in some instances such techniques can be considered to be particularly suited for memory smart cards.

The data integrity of the application data on a smart card may be established if (*e.g.*, only if) all related sectors, including all data blocks, are written completely. In some instances, such as in memory smart cards, security cannot be provided by the smart card itself. Moreover, security is typically a software feature that requires the software to be present in the smart card reader, the Host-PC that controls the reader, or a combination thereof. One instance in which security may be critical is when a smart card is handled by a user at a processing device without physical locking mechanisms because the smart card reader cannot ensure that all sectors and blocks of an application are written completely onto a memory card in such circumstances. Without a physical locking mechanism, the card may move before the write operation has finished. As such, anti-tearing features may be implemented to address data integrity. Techniques may be implemented that for example include splitting existing data groups into different categories such as fixed and dynamic categories, applying anti-tearing to certain categories of the data groups (*e.g.*, apply anti-tearing only to dynamic anti-tearing data groups), applying anti-tearing differently based on subcategories for anti-tearing data groups such as parallel, alternate, and circular, and referencing a valid record list on user media for identifying current or valid data based on anti-tearing. These techniques may minimize the transaction time needed to perform anti-tearing operations while still ensuring data integrity. As a result, the anti-tearing techniques in conjunction with the interoperability features can result in data layouts having data records that can overlap memory block borders with minimal increase in transaction time as compared to the case where no anti-tearing is performed.

FIG. 11 shows illustrative steps involved in configuring interoperable applications with anti-tearing features. At step 1102, an application such as a transit type application may be configured with a data group directory that identifies which available functional data groups of a particular application type are used by the current application of that type. In step 1102, the application may, for example, be configured with a structure that informs host equipment of the identity of which data groups that occupy space on the user

media, which in turn, may also identify the location of different data elements in the data groups because the host equipment may hold pre-existing information on the standard size and formatting for each type of data group or data element for current type of application. At step 1104, data groups may be configured to be divided into different anti-tearing categories. This may be implemented as part of a general specification for applications of the same type or may be individually implemented by application providers. For interoperability, however, it is preferred that anti-tearing categories may be universal to applications of the same type. The categories may for example be fixed and dynamic. At step 1106, one or more of the anti-tearing categories can be divided into subcategories to apply different anti-tearing techniques to different anti-tearing data groups. For example, dynamic anti-tearing data groups may be divided into parallel, alternate, or circular categories. Information about which subcategory is associated with a data group may stored on user media or may part of pre-existing information associated with each data group.

The purpose of anti-tearing is to ensure that data is not corrupted if a transaction is not completed, and to allow minimal inconvenience to the patron as a result of such situations. In situations involving write transactions, where writing can overwrite valid and essential data and where for example, a sequence of blocks belonging together are necessary to retain the logical integrity of that sequence, it may be necessary to implement anti-tearing. As such, within transit type applications, implementation of anti-tearing features for certain data such as the stored value purse (*e.g.*, its references to passes, tickets, usages, and trips) may be necessary. It may also be implemented for example for the status of the application when a hotlist is involved.

In some other instances, anti-tearing protection may not be necessary such as for example for central ticketing initialization and encoding in the context of transit applications. Under this situation, the card can be kept under machine, (or operator) control until the transaction is finished. The machine can be designed so that each transaction can be repeated n times, or reversed, as desired. In the worst case, the offending card can be discarded, and a new one used in its place. If desired, anti-tearing features may also be applied for maintaining data integrity of this type of information depending on system performance capabilities and requirements.

Table 7 below follows the example of Table 6 above to illustrate the categorization of data groups as fixed or dynamic along with examples of approximate byte size used for the data groups. When actually laying out a card format, tables of this type should be taken into consideration preferably so that fixed data groups are laid out as closely

together as possible. Other structural layout confines may be also be applied. For example, it may also be preferred to implement a layout (*e.g.*, a memory smart card layout) in which no fixed and dynamic group should be placed in the same sector together unless anti-tearing protection is provided for them.

Data group name	Is This a Fixed or Dynamic Datagroup	Datagroup-ID	Bytes needed	Write access	Mandatory! Optional
Application identity	Fixed	1	12	once	M
Application condition	Dynamic	2	10	multiple	M
History common application events	Dynamic	3	-	multiple	O
user identity	Fixed	4	10	multiple	Ml
product contracts	Fixed	5	13	multiple	M2
actual products	Dynamic	6	23	multiple	M3
Usage	Dynamic	7	15	multiple	M4
Stored value purse configuration	Fixed	8	6	multiple	M5
Stored value purse	Dynamic	9	6	multiple	M5
Purse event history	Dynamic	10	13	multiple	O

M	Mandatory
Mn	Mandatory for specific conditions
Ml	For personal cards
M2	For identity function for pre-determined products with user related parameters
M3	Always when pre-determined products are stored
M4	Especially when check in / check out leads to fare creation
M5	If value purse is implemented
O	Optional

Table 7

In Table 7, data groups with ID's 1,4, and *5* are typically performed under environments where the card is physically restrained (*e.g.*, ticket vending machines), and/or under the supervision of an individual, and are typically written to only once. Datagroup ID 8 is written to upon the creation of a stored value purse. It is also typically written to once. All dynamic data groups as defined in Table 7 may require anti-tearing protection.

Note that anti-tearing protection could be provided for all data groups regardless of their size, or of the situation with which they are written to if so required. But, for clarity and convenience, only certain data groups such as those that require anti-tearing protection in transit applications are discussed herein to be categorized in a dynamic anti-tearing category.

Table 8 below follows the example of Table 7 to provide an illustrative layout including that includes anti-tearing and interoperability features in an arrangement where

memory size is not a critical consideration. Anti-tearing features are implemented at least by providing replicates for dynamic data groups. Replicates may preferably be provided on a sector to sector basis, as opposed to a data group to data group basis. Data groups are sorted by whether they are fixed or dynamic.

Physical Sector	Appl.- sector	Block	Bytes in Block	Fixed or Dynamic?	Used by data group:
0	-				card application sector
1		Block 0			Application Directory anti tear
		Block 1			valid record list
		Block 2			alternate valid record list
2	Appl.-Sector 0	Block 0	0— 13	fixed	Data group directory
		Block 1	0—5		Data group directory
		Block 1	6— 13		Application identity
		Block 2	0— 1		Application identity
		lock 2	2— 11		User identity
3	Appl.-Sector 1	Block 0	0— 12	fixed	Product contracts
		Block 1	0— 12		Product contracts
		Block 2	0—5		Stored value purse configuration
4	Appl.-Sector 2	Block 0	0—9	dynamic	Application condition Stored value purse
		Block 1	0—5		
		Block 2	0— 13		Actual products
5	Appl.-Sector 3	Block 0	0— 13		Actual products
		Block 1	0— 13	dynamic	Actual products
		Block 2	0—13		Actual products
6	Appl.-Sector 4	Block 0	0— 12	dynamic	Actual products
		Block 1	0— 13		Usage
		Block 2	0— 13		Usage
7	Appl.-Sector 5	Block 0	0— 13		Usage
		Block 1	0— 13	dynamic	Usage
		Block 2	0— 13		Usage
8	Appl.-Sector 6	Block 0	0—4		Usage
9	Appl.-Sector 2	Block 0	0—9		alternate Application condition
		Block 1	0—5		alternate Stored value purse
		Block 2	0— 13		alternate Actual products
10	Appl.-Sector 3	Block 0	0— 13		alternate Actual products
		Block 1	0— 13		alternate Actual products
		Block 2	0— 13		alternate Actual products
11	Appl.-Sector 4	Block 0	0— 12		alternate Actual products
		Block 1	0— 13		alternate Usage
12	Appl.-Sector 5	Block 0	0— 13		alternate Usage
		Block 1	0— 13		alternate Usage
		Block 2	0— 13		alternate Usage
13	Appl. - Sector 6	Block 0	0-4		Alternate Usage

Table 8

The "Application Directory Anti-Tear" may be configured to hold a directory assigning logical sectors to physical sectors. It may be used as a label to identify which sectors contain relevant replicates. However, in many instances, memory use will be an issue. In those

cases, additional anti-tearing features may be applied.

A number of additional anti-tearing features may be implemented for memory-limited cases. For example, subcategories of dynamic anti-tearing data groups may be implemented that may appropriately address anti-tearing concerns and reduce memory usage dedicated to anti-tearing functionality. FIG. 12 shows illustrative steps involved in implementing subcategories of dynamic anti-tearing categories, which may be implemented as part of step 1104 of FIG. 11. The steps of FIG. 12 are presented in a particular order for convenience and clarity. However, the steps may be performed in a different order (*e.g.*, in parallel) or in different combinations. At step 1202, one or more alternate anti-tearing data groups may be implemented. Dynamic data groups that have one record defined (*e.g.*, stored value, application condition) may only need one replicate for data integrity purposes. As such, data groups in the alternate category may be configured with one replicate (*e.g.*, only one replicate) in configuring and executing an application. For example, in the case of an alternate-data group, one logical record set and an alternating record set may be written wherein only one of the record sets is considered valid. In other words, for an alternate data group there may exist only one record in a logical layout, but it may be necessary to implement two records in a physical layout wherein only one will be valid at any given time.

At step 1204, one or more parallel anti-tearing groups may be implemented. Dynamic data groups that define a set of records in parallel, may not have to be doubled. Data groups that may fall in this category in transit applications may include "actual products." Extra memory required for parallel data groups will be dependent on how many valid replicates (*e.g.*, actual products) must be present on the card at any time. Parallel data groups may for example include many record sets that are valid in parallel wherein records are written in random or arbitrary order. For parallel data groups, n records may be valid in parallel. The next record may be written to a free space or to an invalid record that can be overwritten.

At step 1206, one or more circular anti-tearing groups may be implemented. Dynamic data groups that define a set of equal records such as history may not have to be doubled. Data groups that may fall in this category in transit applications may include usage. Extra memory required for circular data groups will be dependent on how many valid replicates (*e.g.*, usages) must be present on the card at any time.

In implementing dynamic data groups such as the subcategories mentioned herein, memory blocks may require reconfiguration as compared to the above-given example in Table 8, which can be performed using the data group directory of an application. In

configuring dynamic data groups, for example for memory cards, it may be preferred to start each record or data group at the beginning of a new block. Thus, a record that exceeds the size of a block size may be followed by a next record that starts at the beginning of a new block. However, it may also be acceptable to fit one or more data groups in one block as long as the data groups together do not exceed the size of the memory block. Circular data groups may for example be configured for history records wherein the next record is written in circular order. For circular data groups, records may be "continuously circulated" based on sequence or some other characteristic.

Correspondingly, by implementing such subcategories, the layout shown in Table 8 may, for example, be changed to that of Table 9 below.

physical Sector	Appl. Sector	Block	Bytes in Block	Data Group Class	Used by data group:
0					card application sector
1		Block 0		alternate	Valid data record list 1
		Block 1		alternate	Valid data record list 2
		Block 2			spare
2	Appl.-Sector 0	Block 0	0— 13	fixed	Data group directory
		Block 1	0— 5		Data group directory
		Block 1	6— 13	fixed	Application identity
		Block 2	0— 1	fixed	Application identity
		Block 2	2— 11	fixed	User identity
3	Appl.-Sector 1	Block 0	0— 12	fixed	Product contracts
		Block 1	0— 12		Product contracts
		Block 2	0—5	fixed	Stored value purse configuration
4	Appl.-Sector 2	Block 0	0—9	alternate	Application condition
		Block 1	0— *5*	alternate	Stored value purse
		Block 2	0—9	alternate	add. Application condition
5	Appl.-Sector 3	Block 0	0— *5*	alternate	add. Stored value purse
		Block 1	0— 13	parallel	1st record Actual products
		Block 2	0—8		1st record Actual products
6	Appl.-Sector 4	Block 0	0— 13		2nd record Actual products
		Block 1	0—8		2nd record Actual products
		Block 2	0— 13		3rd record Actual products
7	AppL-Sector *S*	Block 0	0—8		3rd record Actual products
		Block 1	0— 13		add. 4th record Actual products
		Block 2	0—8		add 4th.record Actual products
8	Appl.-Sector 6	Block 0	0— 13	circular	1st record Usage
		Block 1	0—0		1st record Usage
		Block 2	0— 13		2nd record Usage
9	Appl.-Sector 7	Block 0	0—0		2nd record Usage
		Block 1	0— 13		3rd record Usage
		Block 2	0—0		3rd record Usage
10	Appl.-Sector 8	Block 0	0— 13		4th record Usage
		Block 1	0—0		4th record Usage
		Block 2	0— 13		5th record Usage
11	Appl.-Sector 9	Block 0	0—0		5th record Usage

Table 9

A number of changes exist in Table 9 as compared to the layout of Table 8. For example, the data groups Actual Products and Usage are distributed to always start at a block border. It is assumed that in configuring the layout that three valid Actual Products are to be available at any given time, which leads to including one additional replicate for Actual Products. It is also assumed that in configuring the layout that four valid Usages are to be available at any given time, which leads to including one additional replicate for Usages.

However, each individual application provider is free to alter their requirements for the number of valid data groups and replicates of each type that may be required.

In the example of Table 9, because of the anti-tearing requirements, and the functional requirements of this example specification, some "gaps" in memory space are present, especially in the mapping of the usage data elements. In such circumstances, an application provider is free to subtract (or add) data elements to make more efficient use of the space available.

Various techniques for implementing such features in operation of a smart card or user media may be used. For example, with reference now to FIG. 13, at step 1302, the location of different data such as the location of the value content of different data groups that are associated with a current application may be identified when, for example, user media such as smart cards are read by card readers to provide user with some benefit. As mentioned above, an application provider may configure their application with different data groups selected from a standard list. The pre-existing information about the data groups and information on user media indicating that a current application provider selected certain data groups can be used to identify the specific memory location (*e.g.*, sector, block, offset) of a particular data group. The location of data groups may be variable from application to application because different data groups may have been selected for use in different applications. At step 1304, anti-tearing may applied to a subset of the data on the user media (*e.g.*, to data groups that are in the dynamic class) for the current application. At step 1304, anti-tearing may be applied by exercising steps to identify which data is presently current or valid. Anti-tearing would, for example, identify current data for dynamic data groups even when read/write operation of user media may have been interrupted or incomplete. Step 1304 may include configuring the application in different dynamic categories such alternate, parallel, circular, or combination thereof and applying anti-tearing differently for each category. At step 1306, which may be a sub-step of step 1304, anti-tearing may be applied to only dynamic data groups to the exclusion of other data groups. At step 1308, valid or current data of data groups may be used in the operation of the current application operation based on anti-tearing step 1304 (*e.g.*, to display current values, to use the data for cost, fare, or ticket output, etc.).

Step 1304 may involve generating a valid data record list that is stored on individual user media that is used in anti-tearing operation of system. For example with reference now to FIG. 14, at step 1402, different anti-tearing data groups for data groups providing a function for the current application can be implemented. The application may for

example be of the type mentioned above in connection with step 1302 of FIG. 13. At step 1404, a valid data record list, that is for example configured as a data group on user media for an application, may be read or updated to update or identify valid or most recent information on individual user media. At step, 1404, the location of the valid data record list may be identified using the same techniques as is mentioned above for other data groups and the content may be read or updated to identify or reflect the current state of the value content of corresponding dynamic data groups. As such, the valid data record list may be used to reconcile the validity of different copies of data. The valid data record list may then provide a method for indicating that valid (as well as most recent) data be used. As can be seen form the example of Table 9, which includes a valid data record list, the list can itself be a dynamic data group such as an alternate data group and as such can be exist twice in the layout. The valid data record data list of table 8 would then alternate on each write transaction. The list size may be kept to the block size to assure that it can be written in a single transaction, and can fit in a single block. The purposes of the valid data record list may at least be threefold: providing an anti-tearing mechanism, providing pointers to "most recent" records, and providing a map of "valid" records.

Table 10 below shows examples of validity flags for different classes of dynamic data groups that can be updated in sync with a write sequence switch counter when a write operation is performed for dynamic data.

Write Sequence Switch Counter			
	data class	valid blocks bitmap / Last written	dynamic data group
	alternate	01	stored value purse
	alternate	01	application condition
	parallel	1110	actual products
	circulating	1 out of n (last written)	Usage

Table 10

As shown, the alternate class requires 1 bit for each actual record required, (typically, 2 total bits). If both bits are not set (have a value of zero), then the record was never written. The parallel class may be configured to also require 1 bit for each actual record. For the circular class, the flag may be a pointer to the last record written, and may be set to 0 upon initialization when no record has yet been written. The circular flag may be configured to be incremented up to "n" and decrement down to 1 as appropriate.

In transactions involving non-parallel data group classes, software may be configured so that all new records of all data groups are first written (in an alternate or circular manner). Subsequently, the software which may be part of the current application or may be a separate software application may, for example, write to the alternate position of the valid data record list to set all flags correctly, with the write sequence switch counter incremented. If the write operation to the valid data record list is unsuccessful, the previously written valid data record list, which points to the correct "older" data records.

In transactions involving parallel data classes, existing value in a record may need to be overwritten. To do so, the software may, for example, reset an appropriate bit in the valid data record list to make the record invalid and may alter the write sequence switch counter for this transaction. All other values may be kept unchanged. The valid data record list may then be written to it's alternate position. After that, the software may write all data group records to the appropriate blocks. Lastly, only the validity bits should be flagged accordingly and a second write to the valid data record list may be performed.

FIG. 15 illustratively shows an example of how valid data record lists may be used for user media application. Two valid data record lists 1502 and 1504 may be implemented for alternating use. Valid data record lists 1502 and 1504 follow the example formatting shown in Table 10 above. In this case, valid data record list 1502 may be the current valid data list, for example, because the write switch counter for that list is higher than the write switch counter for list 1504. As shown, data group 2 is an alternate data group having two record sets 1508 for that data group. The value content for data group 2 in list 1502 identifies which one of the two record sets is the currently valid record set. As shown, data group 3 is a parallel data group having parallel record sets 1510. The value content of data group 3 in list 1502 identifies which parallel record sets are valid. As shown, data group 4 is a circular data group having circular record sets 1512. The value content of data group 4 in list 1502 identifies which one of the circular record sets is currently valid. List 1502 and 1504 are absent information for fixed data group 1506 because it is not classified in a dynamic class.

As mentioned above, the valid data record list may preferably be limited to a block size of user media (*e.g.*, 16 bytes) to allow it to be written in one "transaction." There may be typically be two copies of the valid data record list. To recognize the most recently written list, a write switch counter may be included that is increased for each "writing," and which may be cycled. Tables 11, 12, and 13 below provide examples information for valid data record lists.

Valid data record list				
Element Name	Bit Depth	Value Range	Offset in bit	Offset in Byte
WriteSwitchCounter	2	0,1,2	0	0
BitsDataGroupValRecMapEntry	6	$1\text{-}2^6$	2	0.2
ApplTransactionSequence	16	2^{16}	8	1.0
DataGroupValRecMap	88		24	3.0
CRC	16	2^{16}	112	14.0
			128	16.0

Table 11

Element	Description
WriteSwitchCounter	Runs circular from 0 to 2 and back to 0. The list with the follower WriteSwitchCounter is the valid one, *e.g.*, 2 points to "3"
BitsDataGroupValRecMapEntry	DataGroupValRecMap contains valid record information entries for dynamic data groups. The needed bits for one entry of this map is configured by BitsDataGroupValRecMap. The greatest value needed may define the bits needed for all entries. Example: If a data group is configured to contain a maximum of 10 records used in parallel, the needed bits for one entry is 10.
ApplTransactionSequence	This transaction sequence ID is increased by each (writing) transaction for the card application. It is an identity included in transaction data to background systems as a reference to this card transaction regardless of transaction functionality and belonging data groups. This may not be needed for anti-tearing or record validity.
DataGroupValRecMap: DataGroupValRecMapEntry 1 DataGroupValRecMapEntry 2 DataGroupValRecMapEntry 3	This map contains n entries. Each entry has a size of BitsDataGroupValRecMapEntiy. The count of entries = Map size in bits/BitsDataGroupValRecMapEntry. Each entry represents information about valid records of one data group. The sort order of the entries relates to the sort order of "Data group directory entries" which do not belong to fixed data group class. In other words, the "first" DatagroupValRecMapEntry corresponds to the "first" dynamic record.

Table 12

Class of data group	Meaning of DataGroupValRecMapEntiy
alternate: Logically there is only one record needed. For anti-tearing strategy record exists twice and is written alternatively.	2 bits needed. There is always only one bit set. The bit position points to the record of two which is valid. There is the possibility to invalidate both records.
parallel: All records may be in use in parallel. There is no order by design. New record may overwrite any other record decided by content of records.	As many bits needed as there are records of same type defined in data group. Each bit points to one record. Any combination of bits set or reset is allowed.
circular: Data group contains a history. New record is written to next position as last time.	Value is the record pointer to the last written record in circular data group. So this value points to one record exactly. All other records of data group are of older date in circular order. As many bits (n) needed which fit the maximum record counter running from 1 — n (2^n). 0 may be reserved for the initialization.

Table 13

Table 14 provides an example of a valid data record list based on the above-discussion.

Element name	Belongs to data group x sorted by data group directory:	Bit depth	Value	Offset in bit	Offset in Byte
WriteSwitchCounter	-	2	0,1,2	0	0
BitsDataGroupValRecMapEntiy	-	6	8	2	0.2
ApplTransactionSequence I	-	16	0-(2^{16}-l)	8	1.0
DataGroupValRecMap	-				
DataGroupValRecMapEntry 1	2	8	00000001	24	3.0
DataGroupValRecMapEntry 2	9	8	00000001	32	4.0
DataGroupValRecMapEntry3	6	8	00001111	40	5.0
DataGroupValRecMapEntry4	7	8	00000001	48	6.0
Spare		56	0	56	7.0
CRC		16	2^{16}	112	14.0
				128	16.0

Table 14

In this particular example, there are 8 bits for each entry. This allows for a map of 8 invalid/valid record sets in parallel data groups. The bit depth of each DatagroupValRecMap is set by BitsDataGroupValRecMapEntry. The data group "Actual Products" only requires 4 bits, and the pointer for "Last Usage History" only requires three bits (for *5* possible records).

Thus, techniques illustratively described herein can be implemented to provide a format that is secure, efficient, requires minimal transaction time to process, provides anti-tearing protection and meets specific customer needs.

For convenience and clarity, the diagrams in the figures are not illustrated to scale and are not illustrated to appropriate proportional relationship.

The techniques, including the methods and systems, illustratively described herein are sometimes primarily discussed in the context of a transportation application types, transportation services, or specifically a transportation application. However, the techniques may also be application to other forms or types of applications.

Although preferred embodiments of the invention have been described in the foregoing description, it will be understood that the invention is not limited to the specific embodiments disclosed herein but is capable of numerous modifications by one of ordinary skill in the art. It will be understood that the materials used and the chemical details may be slightly different or modified from the descriptions herein without departing from the methods and compositions disclosed and taught by the present invention.